AD-A035 842 NAVAL POSTGRADUATE SCHOOL MONTEREY CALIF F/G 12/2
AN ALGORITHM FOR SOLVING A RANGE CONSTRAINED TRAVELING SALESMAN--ETC(U)
DEC 76 J W HARMS

UNCLASSIFIED NL

| OF |
AD
A035842

END

DATE
FILMED
3-77

(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AN ALGORITHM FOR SOLVING A RANGE CONSTRAINED TRAVELING SALESMAN PROBLEM

by

John William Harms

December 1976

Thesis Advisor: G.T. Howard

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER (9)	
4. TITLE (and Subtitle) (6) An Algorithm for Solving a Range Constrained Traveling Salesman Problem	5. TYPE OF REPORT & PERIOD COVERED Master's Thesis (December 1976)
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) John William Harms	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS (11) Naval Postgraduate School Monterey, California 93940	12. REPORT DATE December 1976
	13. NUMBER OF PAGES 51
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Postgraduate School Monterey, California 93940 (12) 48 p.	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Traveling salesman, vehicle dispatching, branch and bound, integer programming.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A problem of routing earth resource survey aircraft, proposed by NASA, is formulated as a traveling salesman problem, in which the salesman (aircraft) has a range constraint. A heuristic algorithm is presented, which seeks a minimal length set of subtours through n cities. The aircraft begins a subtour at the base location, visits a subset of the n cities and returns when the range constraint prevents a visit to another city. Additional subtours are created until all cities are visited. —> next page

DD FORM 1 JAN 73 1473 (Page 1)

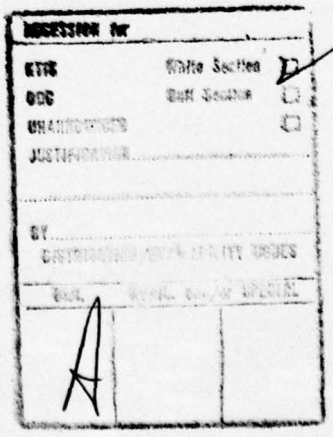
EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

4/B

cont.

The algorithm is programmed in FORTRAN for use on digital computers. The IBM 360/67 computer at the Naval Postgraduate School was used to find solutions to three operational problems of size seven, eighteen and twenty-five cities. Computation times for each problem was under 20 seconds and the solutions were significantly better than feasible solutions calculated without the use of the algorithm.



AN ALGORITHM FOR SOLVING A RANGE CONSTRAINED TRAVELING SALESMAN PROBLEM

by

John William Harms
Captain, United States Army
BS, United States Military Academy, 1969

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the
NAVAL POSTGRADUATE SCHOOL
December 1976

Author: John W Harms

Approved by: Gilbert H. Howard
Thesis Advisor

Second Reader

Chairman, Operations Research Department

David A. Schrady
Dean of Information and Policy Sciences

ABSTRACT

A problem of routing earth resource survey aircraft, proposed by NASA, is formulated as a traveling salesman problem, in which the salesman (aircraft) has a range constraint. A heuristic algorithm is presented, which seeks a minimal length set of subtours through n cities. The aircraft begins a subtour at the base location, visits a subset of the n cities and returns when the range constraint prevents a visit to another city. Additional subtours are created until all cities are visited.

The algorithm is programmed in FORTRAN for use on digital computers. The IBM 360/67 computer at the Naval Postgraduate School was used to find solutions to three operational problems of size seven, eighteen and twenty-five cities. Computation times for each problem was under 20 seconds and the solutions were significiantly better than feasible solutions calculated without the use of the algorithm.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

It is with pleasure that I wish to acknowledge Roger D. Arno, Research Scientist at the National Aeronautics and Space Administration's Ames Research Center, Moffett Field, California for his assistance in formulating this problem and for providing some of the data used in the examples.

I. INTRODUCTION

The National Aeronautics and Space Administration is fostering many technology utilization programs to explore the application of its aerospace technology to other fields. Suggesting how our finite natural resources can be used most effectively is one of its more important considerations. NASA has been working to sophisticate the remote sensing of the earth from a variety of aerospace platforms. The information available from such a program can provide accurate and timely data to assist in the management of the earth's natural and cultivated resources. NASA has demonstrated the usefulness of remote sensing to a wide variety of users with emphasis on the disciplines of geology, agriculture, forestry, oceanology, meteorology and hydrology. For example, an agriculturist can be provided with identification and area measurements of major crop types for use in forecasting potential crop yield. In addition, remote sensing can sometimes detect agricultural diseases and pest attacks before they can be detected from the ground. More detailed discussions of the type of instruments used and examples of the data available from remote sensing can be found in references 4, 6, 7, 13, 15, and 16.

In recent years land has emerged as the one resource "basic to both the good life and the goods of life" [ref 8]. There has been emphasis on efficient land use planning, particularly in metropolitan areas. Many experts are convinced that a well conceived program of remote sensing of metropolitan areas will enhance the quality of urban design. This program could provide, for a given area of land, its

biological productivity, soil limitations, topology, access to transportation and utilities, existing vegetation and the use of adjacent land areas. This data would be a very valuable aid in efficient urban planning.

NASA has been involved in the development of instrumentation to perform the variety of sensing required and demonstration of the data available to potential users. In addition, NASA has been asked by potential users to estimate total program cost and to recommend base locations for a special fleet of aircraft to support a program of remote sensing of urban centers. Assuming an initial cost for the program set up and instrument procurement and a fixed cost associated with each sensing mission, NASA has been seeking methods of analyzing the reduction of operating costs through efficient vehicle routing and effective base locations.

The purpose of this paper is to develop an algorithm which will, given a base location and a set of urban areas to be sensed, provide an optimal or near optimal routing for the number of tours required to visit each of the areas. It is assumed that the operating cost will be primarily a factor of distance traveled plus a fixed charge for each tour required to visit the set of urban areas. Using the fewest flights (subtours) to visit each area and optimally routing the aircraft to insure the minimum distance traveled will yield the minimum operational cost.

This problem was formulated as a multiple traveling salesman problem with the addditional constraint of aircraft range. The aircraft or sensing platform may only be capable of visiting a subset of the set of urban centers before it must return to its base (completing a subtour) to refuel. Thus the problem is to minimize the sum of the distances of the subtours. The algorithm presented by Raymond [ref 14]

was modified to consider this range constraint. The modified algorithm was programmed in FORTRAN and will be discussed along with operational results and recommendations for additional uses.

II. PROBLEM FORMULATION

The traveling salesman problem (TSP) is one of the classic challenges in operations research (OR). The problem is simply stated: a salesman starting from his home wishes to visit a set of cities once and only once and return home. In what order should the cities be visited to minimize the total distance traveled? Mathematical formulation of the problem and a discussion of the constraints can be found in Wagner [ref 17] Chapter 13.

OR literature has included applications of this model to a variety of practical problems. Industrial organizations have used the model to improve the routing of delivery vehicles, an important consideration in light of increased oil prices and rising driver salaries. The "Chinese postman problem" is a similiar problem applied to postal service introduced by the Chinese mathematician Kwan [ref 10]. Municipal governments have found the model valuable for routing garbage collection, street cleaning, meter reading and snow removal vehicles. Other applications include dairy and newspaper delivery. Further discussion of applications can be found in Bradley's "Survey of Deterministic Networks" [ref 3].

The key difference in the classic statement of the TSP and the remote sensing problem is the addition of a range constraint. The salesman of the TSP may refuel during his trip without altering the optimal route. This is not feasible for the aircraft performing remote sensing. There exists a point during the trip when it must return to its base to refuel and then continue the mission. Therefore,

the problem statement becomes: an aircraft starting from its base must visit a set of urban areas once and only once and return to its base. The aircraft is subject to a range limitation for any given flight; that is, it may only be capable of visiting a subset of the set of cities before it must return to its base to refuel. Each time the aircraft returns to its base a subtour is completed and additional flights (subtours) are created as necessary to accomplish a visit to each city. The objective is to minimize total distance traveled as the sum of the length of each subtour.

A subtour, t, can be represented as a set of ordered city pairs, e.g.

$$t = [(i_1,i_2),(i_2,i_3) \ldots (i_{n-1},i_n),(i_n,i_1)]$$

where i_1 is always the base. The mathematical formulation of the problem is to find the set of subtours to:

$$\text{minimize} \quad \sum_{k=1}^{s} l_k \tag{2.1}$$

$$\text{subject to} \quad l_k = \left[\sum_{j=1}^{n_k} d_{i_j,i_{j+1}}\right] + d_{i_{n_k+1},i_1} \tag{2.2}$$

$$l_k \leq r \qquad k=1,2,\ldots,s \tag{2.3}$$

$$1 \leq s \leq nc \tag{2.4}$$

each city in exactly one subtour

where s = number of subtours

l_k = length of subtour k

n_k = number of cities in subtour k

d_{ij} = distance from city i to city j

r = range of aircraft

nc = number of cities to be visited

Note that if s=1, eq. 2.2 replaces eq. 2.1 and the problem is a standard TSP. The upper bound of eq. 2.4 represents the worse case situation that each city must be visited by a seperate subtour. Eq. 2.3 represents the range constraint.

The distance matrix (D_{ij}) represents the euclidean distance from city i to city j. During actual sensing missions the aircraft must fly several passes over a city to completely image the area. This distance (loiter distance) depends on three factors: the size of the area, the percent side overlap used for film images and the turning radius of the aircraft. Each city will have a unique loiter distance (ld_i) that must be considered when the subtour length is calculated. When the problem is formulated the value ld_i is added to each element in column i. Thus the new distance matrix is $D'_{ij} = D_{ij} + ld_i$ for all i. The new matrix is used to compute subtour lengths.

This problem appears similiar to two existing expansions

of the TSP; the multisalesman problem (MTSP) and the vehicle dispatching problem. In the MTSP, m salesmen starting from a base city must visit n cities and return. Each city is visited by one and only one of the m salemen. The objective is to route each of the m salesmen such that the total distance traveled is minimized. Bellmore and Hong [ref 2] provide a detailed discussion and formulation of this problem. The objective of the vehicle dispatching problem is to obtain delivery routes from a depot to a set of demand points for a fleet of vehicles such that the total distance traveled by the entire fleet is minimized as with the m traveling salesmen. The problem is enhanced by finite vehicle capacities, route time constraints and varying quantity demanded at each demand point. A complete description cf this type application can be found in Golden [ref 12].

There are important differences between the remote sensing problem and the applications mentioned above. The MTSP has no upper bound on the number of cities any one of the m salesmen can visit. The aircraft range constraint requires consideration of a upper bound on the number of cities that can be visited during any given flight. The vehicle dispatching problem is concerned with routing each vehicle in the fleet. Although potential users will presumably have a "fleet" of aircraft to accomplish its sensing missions, not every aircraft will be scheduled to sense a subset of the areas to be visited. For example, given a fleet of three aircraft and the determination that two subtours are required to visit the set of cities, only two aircraft (or one aircraft making two flights) will be scheduled. This decision assumes the fixed cost charged each time an aircraft is launched is sufficiently high to make it more desireable to create the fewest number of flights.

The remote sensing problem, therefore, is not a simple restatement of an existing TSP. The addition of a range constraint requires special considerations. The solution technique must carefully create only the number of subtours required and route each subtour so that the total distance traveled is minimized.

III. SOLUTION ALGORITHMS

One of the most intriguing aspects of the TSP is, that although simply stated, it remains unsolved in a closed form. The computational difficulty with known solution tecnniques is the exponential increasing solution time as the size of the problem increases. There are two types of solution algorithms: heuristic and exact. The latter yields the optimal tour. The difficulty with this type algorithm is the large computer storage and time requirements which are related to the large number of possible solutions. For example, given a symmetric TSP there are (n-1)!/2 possible solutions (n=number of nodes). A relatively small problem of ten nodes yields 181,440 solutions. Fortunately, many applied or real world problems do not demand the optimal solution. Powerful heuristic algorithms have been developed which will yield near optimal solutions and provide a substantial savings in computer storage, time and money. An excellent survey of existing solutions is given by Bellmore and Nemhauser [ref 2].

The Bellmore and Nemhauser article classifies the algorithms by solution generation method. There are three fundamentally different ways of generating solutions: tour-to-tour improvement, tour building, and subtour elimination. The remote sensing problem requires tour length feasibility be maintained as the tour is formed. That is, if visiting the next city (node) will result in violating the range constraint, a new subtour is started. Consequently, a tour building solution was considered the best technique to adapt to this problem.

Both exact and heuristic algorithms were studied for applicability. A literature survey provided a number of both types of algorithms. An exact solution considered was presented by Christofides and Eilon [ref 5]. Their paper discusses three solution methods to the vehicle-dispatching problem. The branch and bound approach, a modification of the algorithm published by Little et. al. [ref 11], would reach an exact solution to this problem provided the number of subtours required was known a priori. If s (the number of subtours) was known, the base location would be replaced by s artificial bases as explained in ref 5. The algorithm would progress using the branch and bound method to insert nodes into one of the tours. Before branching to a new node, it would be necessary to check that the total distance accumulated on each subtour did not exceed the range. The key to this algorithm is knowing the number of subtours a priori.

As noted in the problem formulation section, there exists an upper bound on the number of subtours. Therefore, it is feasible to select an s known, a priori to be sufficiently large, and apply this branch and bound algorithm. The solution would be recorded, the number of subtours decreased by one and a new soluton computed. This procedure would continue until there are insufficient subtours to complete a visit to each city and the last recorded solution would be the optimal. The success of this procedure would be very dependent upon the initial selection of s. A poor selection could involve many iterations and would require considerable user interaction and computer time. This would be particularly undesireable for problems that require a solution in a reasonably quick time. (Problems of this type are discussed in Section VI). Therefore, application of this algorithm was not considered profitable.

Limiting the study of heuristic algorithms to those using the tour building generation technique, the algorithm presented by Raymond [ref 14] was selected to modify and apply to this problem. The approach of Raymonds's algorithm is similiar to that of Karp and Thompson [ref 9] and will obtain optimal or near optimal solutions to the classical TSP. The algorithm successively inserts nodes into the tour by using a series of decision rules and provides for an exchange procedure to determine if the total distance traveled may be reduced by rearranging the order in which the nodes are visited. A detailed explanation of the algorithm steps is available in ref 14.

The modified algorithm starts by creating a subtour from the base to the most distant city. The same criteria as the original algorithm is used to select the next node for insertion. That is, increment calculations (I_k) are made for each remaining node (k). This calculation is made by inserting the node k between each pair of nodes (p,q) in the subtour and computing the incremental increase in subtour length, $I_k = d_{pk} + d_{kq} - d_{pq}$. Each node will have a minimum increment calculation (I^*_k) and a next best increment calculation (J^*_k). If the number of cities in the tour is less than four the maximum I^*_k determines the node k to be inserted between the nodes that produced I^*_k. Otherwise, the node k for which $J^*_k - I^*_k$ is a maximum is selected and inserted between the nodes that produced I^*_k. Before completing the insertion, the algorithm looks ahead to determine if the range constraint will be violated. If it is not violated the node is inserted and the three link exchange procedure described by Raymond is applied. This

procedure takes each node in turn out of its present subtour, reevaluates its insertion increments in that subtour and inserts the node between the nodes that now give the minimum tour length increment. If the constraint is violated the node is not inserted and an additional subtour is formed.

The nearest city to the base not yet in a subtour is used to form the next subtour. The criteria for inserting nodes for the multiple tour problem is changed. A node is inserted in the subtour that will create the least increase in total distance (sum of subtour lengths). A procedure was developed that exchanges nodes among subtours to determine if a savings in total distance traveled is possible by placing a node in a different subtour. Before any node insertions or exchanges among subtours is completed, the algorithm looks ahead to insure tour length feasibility is maintained. The algorithm progresses iteratively until each city is visited.

IV. THE ALGORITHM

The steps of the algorithm:

1. With the base as one node select the node farthest from base (assumed to be j*) and form a tour with links (B,j*) and (j*,B). Set the number of subtours s equal to 1 and the number of links l_1 equal to 2.

2. Define I^*_k as the minimum tour-length increment that results from the insertion of node k in each existing link (p,q). Define nodes p* and q* as the nodes connected by the link (p*,q*) that produced I^*_k. In addition define J^*_k as the increment for the node k most nearly equal to I^*_k. Thus for node k, I^*_k is the best increment, J^*_k the next best increment and (p*,q*) the link to break if inserting node k. Calculate I^*_k and J^*_k for each node that is not currently included in the tour.

3. If $l_1 \leq 4$ choose the node m for which I^*_k is a maximum. If $l_1 > 4$ choose the node m for which $J^*_k - I^*_k$ is a maximum.

4. Insert the node m into the tour between p* and q* by adding links (p*,m) and (m,q*) and deleting link (p*,q*).

Calculate the new tour distance d_1. If $d_1 \leq$ range, complete the insertion and increase the number of links in the tour by one. If $d_1 >$ range, do not insert node m and go to step 7 to create a new subtour.

5. Take each node in turn out of the present tour, reevaluate its insertion increments and insert the node between the pair of nodes that now gives the minimum tour length increment. (In most cases the node is reinserted in the link created when the node was taken out of the tour, producing no change.)

6. If all nodes have been added to the tour stop, otherwise go to step 2.

7. Select the node m*, not already in a subtour, closest to the base and form a tour with links (B,m*) and (m*,B). Set the number of subtours s equal to s+1 and the number of links l_s equal to 2.

8. Calculate I^*_k as in step 2 for each node k remaining to be added to a subtour. Consider the subtours in the order in which they were created. For each subtour select the node that produced the minimum I^*_k. Before inserting the node two checks are made. First insure that the range constraint will not be violated. Second insure that the total distance traveled (sum of subtour lengths) will not be less if the selected node is inserted in a different subtour. Only those nodes which pass these checks are inserted. Thus at any given iteration a node need not be inserted in each subtour. A given node may have a smaller

I^*_k in one subtour, but if inserted, would force range constraint violation. In this situation place the node in the next best subtour. If nodes may not be added to any existing subtour without violating the range constraint go to step 7.

9. Repeat step 5 for each subtour to which a node was added.

10. For each subtour make the increment calculations using the nodes from the other subtours. Select the minimum I^*_k and check if the total distance traveled may be reduced by adding the selected node to the given subtour and deleting it from its original subtour. If the distance is reduced make the change and repeat step 5.

11. If all nodes are contained in a subtour stop, otherwise go to step 8.

V. TEST PROBLEMS

The computer program was used to provide routes from bases at Moffett Field, California, Kansas City and Wallops Station, Maryland to fifty selected urban centers. Figure 1 is the grid system used to calculate the distances between cities. The cities were placed in one of three sections by the following criteria: the straight lines from Moffett Field to Kansas City and Wallops Station to Kansas City were bisected by a perpendicular, as shown in Figure 1, creating three sections. The cities would be imaged from the base in their section. Table 1 provides the coordinates and loiter distance of each city. The range of the aircraft was assumed to be 2900 statute miles. Tables 2-1, 2-2 and 2-3 are the routing recommendations of the computer program.

Tables 3-1, 3-2 and 3-3 are the results of a feasible solution to the same problem computed at NASA's Ames Research Center. The constraints observed for determining this solution were: the aircraft would have a range of 2900 statute miles, no more than three urban centers would be imaged during any subtour and flights from Moffett Field and Wallops Station would be held to a reasonable minimum.

The NASA solution required 29 subtours and a total distance of 52,660 statute miles. The algorithm solution constructed 15 subtours with a total distance of 36,505 statute miles traveled. A savings of 16,155 statute miles results from the algorithm solution. This large savings is due, in part, to the differing criteria used to formulate the solutions. However, it does show the usefulness of the algorithm and the speed at which the program can compute a final solution.

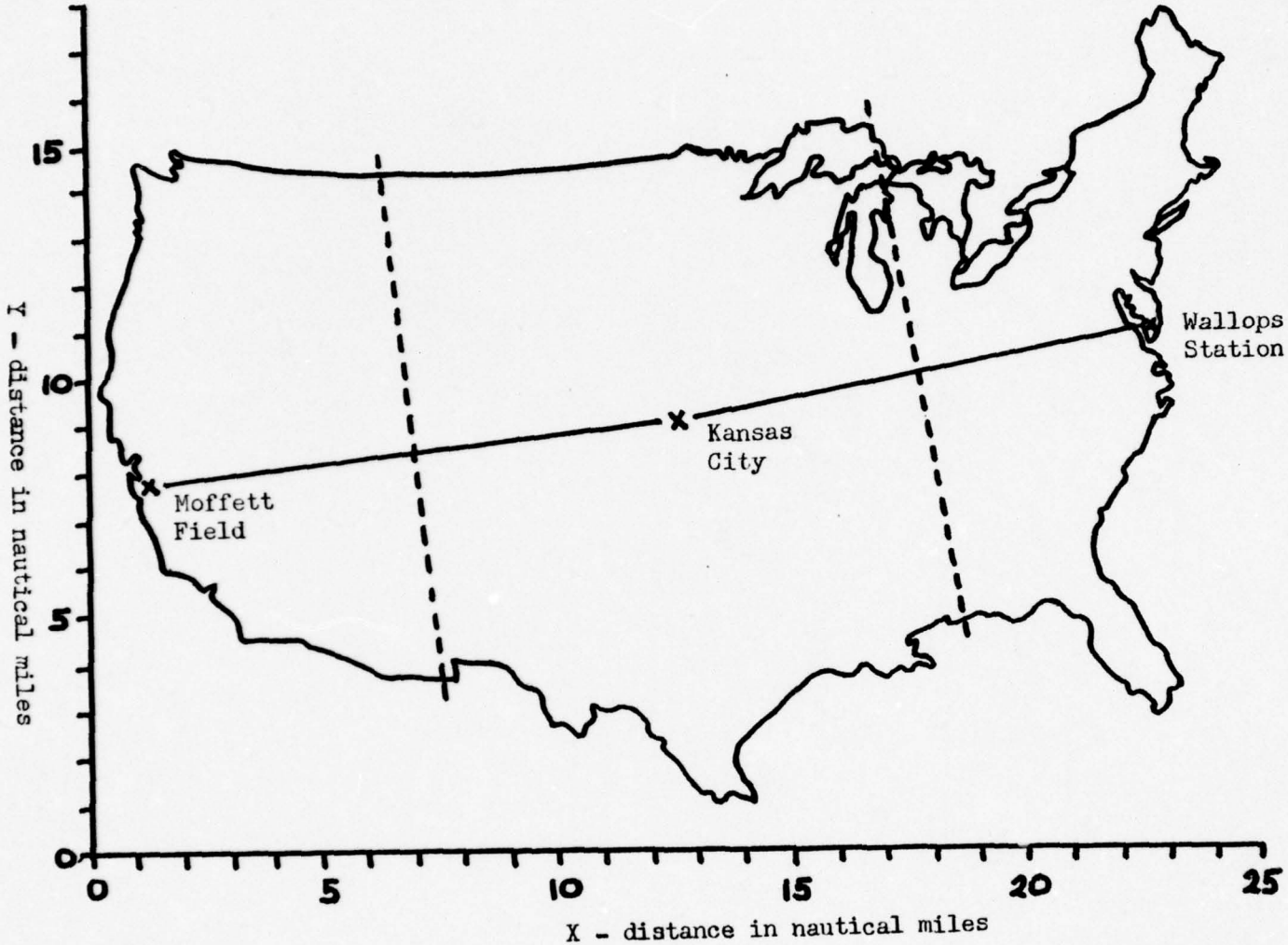


FIGURE 1

X-Y COORDINATE SYSTEM

TABLE 1

CITY COORDINATES AND LOITER DISTANCE

City	X	Y	Loiter Distance
Albany	2230	1390	190
Allentown	2190	1230	190
Atlanta	1950	700	290
Baltimore	2190	1150	340
Birmingham	1820	650	340
Boston	2350	1470	280
Buffalo	2030	1310	240
Chicago	1660	1120	630
Cincinnati	1830	1015	350
Cleveland	1930	1160	410
Columbus	1900	1080	230
Dallas	1350	500	740
Dayton	1820	1080	250
Denver	910	900	450
Detroit	1850	1230	350
Greensboro	2150	910	290
Hartford	2300	1350	150
Houston	1470	350	670
Indianapolis	1730	1030	400
Kansas City	1380	880	380
Los Angeles	220	560	670
Louisville	1780	910	190
Memphis	1560	710	220
Miami	2300	370	370
Milwaukee	1620	1210	270

[1] Loiter distances provide by Ames Research Center.

TABLE 1 (con'd)

CITY COORDINATES AND LOITER DISTANCE

City	X	Y	Loiter Distance
Minneapolis	1400	1260	380
Nashville	1780	800	270
New Orleans	1740	410	290
New York	2300	1300	730
Norfolk	2270	2020	230
Oklahoma City	1290	660	330
Omaha	1300	1010	230
Philadelphia	2230	1200	480
Phoenix	560	470	250
Pittsburgh	2030	1150	430
Portland	150	1250	450
Providence	2360	1370	160
Rochester	2090	1330	150
Sacramento	150	850	410
St Louis	1590	910	500
St Petersburg	2140	410	250
Salt Lake City	580	910	200
San Antonio	1270	270	290
San Bernardino	310	560	190
San Diego	320	480	510
San Francisco	100	780	530
Seattle	190	1420	500
Syracuse	2160	1350	310
Toledo	1830	1150	230
Washington DC	2200	1100	350
Moffett Field	120	760	
Kansas City	1380	880	
Wallops Station	2270	1080	

TABLE 2-1

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: MOFFETT FIELD

Subtour	Cities Imaged	Length
1	Sacramento - Seattle - Portland	2892
2	Los Angeles - San Diego - San Bernadino	2184
3	San Francisco - Salt Lake City - Phoenix	2696

Total Distance = 7772 miles

Time to Compute Solution= 1.27 seconds

TABLE 2-2

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: KANSAS CITY

Subtour	Cities Imaged	Length
1	Houston - San Antonio - Oklahoma City	2878
2	Dallas - Denver	2853
3	Omaha	581
4	Indianapolis - Louisville - St. Louis - Kansas City	2520
5	Nashville - Birmingham - New Orleans - Memphis	2746
6	Minneapolis - Milwaukee - Chicago	2514

Total Distance = 14092

Time to Compute Solution = 6.97 seconds

TABLE 2-3

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: WALLOPS STATTION

Subtour	Cities Imaged	Length
1	St Petersberg - Miami - Norfolk	2642
2	Columbus - Cincinnati - Dayton - Detroit - Buffalo	2816
3	Hartford - Providence - Boston - Albany - Philadelphia	2401
4	New York - Syracuse - Rochester - Allentown - Baltimore	2607
5	Washington, DC - Pittsburgh - Cleveland - Toledo	2451
6	Greensboro - Atlanta	1724

Total Distance = 14641

Time to Compute Solution = 17.02 seconds

TABLE 3-1

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: MOFFETT FIELD

Subtour	Cities Imaged	Length
1	Salt Lake City	1950
2	Phoenix	2340
3	Denver - Omaha	1990
4	Houston	2020
5	San Antonio	1990
6	Dallas	1990
7	Oklahoma City - Kansas City	1410
8	New Orleans - Memphis	1960
9	St Louis - Minneapolis	1850
10	Chicago - Milwaukee	1800
11	Miami	2780
12	St Petersburg	2250
13	Birmingham - Atlanta	2070
14	Nashville - Louisville - Indianapolis	1950
15	Cincinnati - Dayton - Columbus	2100
16	Detroit - Toledo	1880
17	Cleveland - Pittsburgh	2380
18	Buffalo - Rochester	2190
19	Syracuse - Albany	2710

Total Distance = 39520 miles

TABLE 3-2

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: KANSAS CITY

Subtour	Cities Imaged	Length
1	Seattle	1950
2	Portland	1450
3	San Francisco - Sacramento	1140
4	Los Angeles - San Bernadino	1510
5	San Diego	1400

Total Distance = 7450 miles

TABLE 3-3

FLIGHT MISSIONS FOR URBAN CENTER COVERAGE
BASE: WALLOPS STATION

Subtour	Cities Imaged	Length
1	Boston - Providence - Hartford	1430
2	New York	1120
3	Philadelphia - Allentown	1080
4	Washington, DC - Baltimore	970
5	Norfolk - Greensboro	1090

Total Distance = 5690 miles

VI. ADDITIONAL APPLICATIONS

The purpose of this paper was to develop an algorithm that will provide optiimal or near optimal routing for a range constrained vehicle visiting a set of nodes. The algorithm may be used to solve similiar problems. Several considerations for use of this algorithm are discussed in this section. They are: under-flying satellite coverage, isolated disaster coverage and point reconaissance.

Satellites are often used to image large land masses and bodies of water. These orbiting platforms perform the sensing mission during periodic passes over the critical area. The product of a mission may sometimes be impaired by cloud cover or may reveal the need for finer resolution imagery. The areas requiring additional coverage are assumed dispersed and relatively small. These areas can be considered as the cities of the original problem and the algorithm used to determine, from a given base, the number of flights required, the optimal or near optimal routing and the total distance traveled.

Disasters such as tornados or flooding rivers may require government agencies to estimate damage and make recommendations for appropriating the relief effort. Remote sensing can provide valuable data to assist in the decision process. Consider a flooding river; it is likely several areas along the flood path need immediate attention. The goal is to provide the data for evaluation as rapidly as possible. The algorithm can provide routing from a base to the areas of concern such that the sensing mission is completed in the shortest possible time. It is assumed that

least distance routing implies the least time will be required to accomplish the mission.

Several other applications are similiar to the disaster coverage and involve routing reconnaissance aircraft. For example, when fighting large forest fires, "hot spots" develop that need periodic monitoring. Also, in a large theater of operations, the military commander often requires timely reconnaissance of key objective areas. In both these situations the goal is to provide the information to the user as rapidly as possible. Therefore, the algorithm can be used to route the aircraft to the critical areas and provide the information as quickly as possible.

VII. CONCLUSIONS

The algorithm and computer program presented can successfully construct an optimal or near optimal solution to the range constrained vehicle routing problem formulated in this paper.

Several aspects of remote sensing, considered beyond the scope of this paper, provide a basis for further study in this area. The angle of the sun can have an appreciable effect on the resolution of the imagery. Sun-angle is a function of the time of day and could be considered in the computer program by adding a time dimension. Using aircraft speed, it would be possible to calculate a time period during which the aircraft would image a given city. This would provide the user the ability to insure that the desired sun-angle occurs during the imaging.

This paper assumes each aircraft has sufficient sensor capacity to image any number of cities in a subtour. It is likely there may be a finite capacity for certain sensors. For example, assume only a fixed amount of film can be placed on the aircraft. The film requied to image all the cities in a given subtour may exceed this amount. To consider this problem, the aircraft would start with a finite amount of film which would be reduced by the amount required to image a given city as that city is visited. Thus sensor capacity could be added as a constraint and considered in the same manner as the range constraint.

The problem of imaging areas as widely dispersed as the fifty urban centers listed in Table 1 requires multiple

bases. The example problem assigned cities to bases by section. This approach is reasonable, however, it would be preferable to permit the computer program to make this assignment. One consideration is to construct an artificial base from which all flights originate and sufficient dummy bases at the same location as the actual bases to permit multiple subtours through that base. The distance matrix must be constructed such that each subtour starts from the artificial base to one of the actual bases and is routed back through the actual base to the artificial base. Formulation of a fifty city problem with artificial and dummy bases could require considerable computer storage.

APPENDIX A

COMPUTER PROGRAM

```
*****************************************************************
*                    VARIABLE DEFINITIONS                       *
*                                                               *
*                                                               *
*   X(I)=X COORDINATE OF THE I-TH CITY                          *
*                                                               *
*   Y(I)=Y COORDINATE OF THE I-TH CITY                          *
*                                                               *
*   Z(I)=LOITER DISTANCE OF THE I-TH CITY                       *
*                                                               *
*   D(I,J)=DISTANCE MATRIX                                      *
*                                                               *
*   RANGE=AIRCRAFT RANGE                                        *
*                                                               *
*   NTOURS=NUMBER OF SUBTOURS CREATED                           *
*                                                               *
*   LINKS(I)=NUMBER OF LINKS IN SUBTOUR I                       *
*                                                               *
*   INNODE(I,J)=VECTOR OF CITIES IN SUBTOUR I                   *
*                                                               *
*   NONODE(I)=VECTOR OF CITIES TO BE VISITED                    *
*                                                               *
*   LENGTH(I)=LENGTH OF SUBTOUR I                               *
*                                                               *
*   TOTDIS=TOTAL DISTANCE TRAVELED                              *
*                                                               *
*   LH(I,J)=LEFT HAND ENDPOINT OF SUBTOUR I, LINK J             *
*                                                               *
*   RH(I,J)=RIGHT HAND ENDPOINT OF SUBTOUR I, LINK J            *
*                                                               *
*   NC=NUMBER OF CITIES IN THE PROBLEM                          *
*                                                               *
*   S(I,J)=IN SUBTOUR I THE SUCCESSOR OF CITY J                 *
*                                                               *
*   PRED(I,J)=IN SUBTOUR I THE PREDECESSOR OF CITY J            *
*                                                               *
*   STAR(I,J)          >                                        *
*   NNSTR(I,J)         >                                        *
*   LLSTR(I,J)         >                                        *
*   SECSTR(I,J)        >  STORAGE MATRICES FOR ALGORITHM        *
*   SNNSTR(I,J)        >  CALCULATIONS                          *
*   SLLSTR(I,J)        >                                        *
*   THETA(I,J)         >                                        *
*                                                               *
*                                                               *
*****************************************************************
```

```
C    MAIN

      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,KK,
     1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
      DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
      INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
      REAL LENGTH
```

```
C    INITIALIZE ALGORITHM VARIABLES

       CALL INIT
       IPRINT=0
       IF(ISTOP .NE. 0) GO TO 100

C    THE CRITERIA TO FORM THE FIRST SUBTOUR IS TO SELECT THE
C    CITY FURTHEST FROM THE BASE AS THE FIRST CITY TO VISIT

       AMAX=-1.0
       DO 17 J=1,NC
          IF(D(1,J) .GT. AMAX) GO TO 18
          GO TO 17
    18    AMAX=D(1,J)
          NODE=J
    17 CONTINUE

C    FORM THE FIRST SUBTOUR

       S(1,1)=NODE
       S(1,NODE)=1
       LINKS(1)=2

C    ESTABLISH ENDPOINTS

    53 CALL ENDPTS

C    IF USER DESIRES TO PRINT SUCCESSOR FUNCTION AT EACH
C    ITERATION SET IPRINT=3

       IF(IPRINT .NE. 3) GO TO 54
        DO 109 I=1,NTOURS
          WRITE(6,108) (S(I,J),J=1,NC)
   108    FORMAT(2513)
   109 CONTINUE

C    CALCULATE TOUR LENGTHS AND TOTAL DISTANCE

    54 CALL TORDIS

C    ESTABLISH NONODE AND INNODE VECTORS

       CALL TOVIST

C    WHEN IEND=0 ALL CITIES HAVE BEEN VISITED

       IF(IEND .EQ. 0) GO TO 50

C    SELECT CRITERIA TO DETERMINE NEXT CITY TO VISIT

   999 STOP
       IF(NTOURS .GT. 1) CALL GRATOR
       IF(NTOURS .EQ. 1) CALL SNGLTR
       CALL ENDPTS
       CALL THREEX
       GO TO 53

   100 WRITE(6,105) ISTOP
   105 FORMAT(10X,'CITY',I5,' IS OT OF RANGE')
       GO TO 999
    50 DO 110 I=1,NTOURS
          WRITE(6,111) (S(I,J),J=1,NC)
   111    FORMAT(2513)
   110 CONTINUE
       DO 112 I=1,NTOURS
          WRITE(6,113) I,LENGTH(I)
   113    FORMAT(10X,'LENGTH OF SUBTOUR',I3,' =',F15.4)
   112 CONTINUE
       WRITE(6,115) TOTDIS
   115 FORMAT(10X,'TOTAL DISTANCE',F20.4)
       END
```

```
      SUBROUTINE INIT
C     SUBROUTINE WILL READ PROBLEM SIZE, CITY LOCATIONS AND
C     LOITER DISTANCES, AIRCRAFT RANGE, INITIALIZE SUCCESSOR
C     AND PREDECESSOR FUNCTIONS AND COMPUTE DISTANCE MATRIX
      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,KK,
     1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
      DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
      INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
      REAL LENGTH

      IPRINT=0
      ISTOP=0
      NTOURS=1
C     READ NUMBER OF CITIES IN PROBLEM AND AIRCRAFT RANGE
      READ (5,11) NC,RANGE
   11 FORMAT(I3,F20.4)
      HALFR=RANGE/2.
C     READ CITY LOCATIONS AND LOITER DISTANCES, BASE MUST BE
C     READ AS CITY 1
      DO 20 I=1,NC
         READ(5,21) X(I),Y(I),Z(I)
   21    FORMAT(3F10.4)
   20 CONTINUE
C     COMPUTE DISTANCE MATRIX
C     THE FACTOR 1.15 CONVERTS NAUTICAL MILES TO STATUTE MILES
      DO 12 I=1,NC
      DO 13 J=1,NC
         D(I,J)=(SQRT(((X(I)-X(J))**2+((Y(I)-Y(J))**2))))*
         1.15+Z(J)
C     CHECK FOR CITY OUT OF RANGE
         IF(I.EQ.1) GO TO 14
         GO TO 13
   14    IF(D(I,J) .GT. HALFR) ISTOP=J
   13 CONTINUE
   12 CONTINUE
C     INITIALIZE SUCCESSOR AND PREDECESSOR FUNCTION
      DO 15 I=1,NC
      DO 16 J=1,NC
         S(I,J)=0
         PRED(I,J)=0
   16 CONTINUE
   15 CONTINUE
C     IF USER DESIRES TO PRINT DISTANCE MATRIX SET IPRINT=1
      IF(IPRINT .NE. 1) GO TO 30
      DO 150 I=1,NC
         WRITE(6,106) (D(I,J),J=1,NC)
  106    FORMAT(25F4.0)
  150 CONTINUE
   30 RETURN
      END
C
      SUBROUTINE ENDPTS
```

```
C      SUBROUTINE ESTABLISHES, FOR EACH SUBTOJR, THE
C      ENDPOITS OF EACH LINK

       COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
      1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,KK,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
      1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
      1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH

       IPRINT=0
       DO 22 J=1,NTOURS
          LH(J,1)=1
          RH(J,1)=S(J,1)
          LIMIT=LINKS(J)
       DO 23 K=2,LIMIT
          IM=K-1
          IL=RH(J,IM)
          LH(J,K)=RH(J,IM)
          IR=LH(J,K)
          RH(J,K)=S(J,IR)
    23 CONTINJE
    22 CONTINUE

C     IF USER DESIRES TO PRINT THE ENDPOINTS OF EACH LINK AT
C     EVERY ITERATION SET IPRINT=4

       IF(IPRINT .NE. 4) GO TO 25
       DO 110 I=1,NTOURS
          LIMIT=LINKS(I)
       WRITE(6,102) (LH(I,J),RH(I,J),J=1,LIMIT)
   102 FORMAT(5X,'ENDPOINT',2I10)
   110 CONTINJE
    25 RETURN
       END

       SUBROUTINE TORDIS

C      SUBROUTINE CALCULATES LENGTH OF EACH SUBTOUR AND TOTAL
C      DISTANCE AS SUM OF SUBTOUR LENGTHS

       COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
      1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
      1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
      1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH

       IPRINT=0
       TOTDIS=0.0
       DO 55 I=1,NTOURS
       NS=S(I,1)
       NP=1
          LENGTH(I)=0.0
    20 LENGTH(I)=D(NP,NS)+LENGTH(I)
          NP=NS
          NS=S(I,NP)
          IF(NP .GT. 1) GO TO 20
          TOTDIS=TOTDIS+LENGTH(I)
    55 CONTINUE

C     IF USER DESIRES TO PRINT LENGTH OF EACH SUBTOUR AND
C     TOTAL DISTANCE AT EVERY ITERATION SET IPRINT=5
```

```
      IF(IPRINT .NE. 5) GO TO 25
      DO 105 I=1,NTOURS
         WRITE(6,102) I,LENGTH(I)
  102    FORMAT(10X,'LENGTH OF SUBTOUR',I3,' =',F15.4)
  105 CONTINUE
      WRITE(6,101) TOTDIS
  101 FORMAT(10X,'TOTAL DISTANCE',F20.4)
   25 RETURN
      END
C
      SUBROUTINE TOVIST

C     SUBROUTINE ESTABLISHES A VECTOR (NONODE) OF THE CITIES
C    THAT HAVE NOT BEEN VISITED AND FOR EACH SUBTOUR A VECTOR
C    (INNODE) OF CITIES IN THAT SUBTOUR


      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
     1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
      DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
      INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
      REAL LENGTH


      KK=0
      IEND=0
      DO 24 J=2,NC
      DO 25 I=1,NTOURS
         IF(S(I,J) .NE. 0) GO TO 24
         IF(I .LT. NTOURS) GO TO 25
         IEND=1
         KK=KK+1
         NONODE(KK)=J
   25 CONTINUE
   24 CONTINUE
      DO 26 I=1,NTOURS
      MM=1
      DO 27 J=2,NC
         IF(S(I,J) .EQ. 0) GO TO 27
         INNODE(I,MM)=J
         MM=MM+1
   27 CONTINUE
   26 CONTINUE
      RETURN
      END

      SUBROUTINE SNGLTR

C     SUBROUTINE PLACES NEXT CITY IN SUBTOUR


      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
     1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
      DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
      INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
      REAL LENGTH


      CALL TNCRMT
      AMAX=99999.
      AMIN=-1.0
      IPASS=0
      IF(LINKS(1) .GT. 4) GO TO 34
      LIMIT=LINKS(NTOURS)
   28 DO 29 J=1,KK
      DO 30 I=1,LIMIT
```

```
      IF(THETA(I,J) .LT. AMAX) GO TO 31
      GO TO 30
31    NN=NONODE(J)
      LL=I
      AMAX=THETA(I,J)
30 CONTINUE
      STAR(1,J)=AMAX
      NNSTR(1,J)=NN
      LLSTR(1,J)=LL
      AMAX=99999.
29 CONTINUE
   DO 32 K=1,KK
      IF(STAR(1,K) .GT. AMIN) GO TO 33
      GO TO 32
33    IN=NNSTR(1,K)
      INL=LLSTR(1,K)
      AMIN=STAR(1,K)
32 CONTINUE
   IF(IPASS .EQ. 1) GO TO 35
   CHECK=LENGTH(1)+AMIN
   IF(CHECK .GT. RANGE) CALL TOURNU
   IF(CHECK .GT. RANGE) GO TO 53
   IL=LH(NTOURS,INL)
   IR=RH(NTOURS,INL)
   S(NTOURS,IL)=IN
   S(NTOURS,IN)=IR
   LINKS(1)=LINKS(1)+1
   GO TO 53


34 IPASS=1
   GO TO 28
35 LIMIT=LINKS(NTOURS)
   AMAX=99999.
   DO 36 I=1,KK
   DO 37 J=1,LIMIT
      IF(THETA(I,J).GT.STAR(1,I).AND.THETA(I,J).LT.AMAX)
  1   GO TO 38
      GO TO 37
38    AMAX=THETA(I,J)
      LNK=J
      NOD=NONODE(KK)
37 CONTINUE
      SECSTR(1,I)=AMAX
      SNNSTR(1,I)=NOD
      SLLSTR(1,I)=LNK
      AMAX=99999.
36 CONTINUE
      AMIN=-1.
   DO 39 K=1,KK
      CRIT(K)=SECSTR(1,K)-STAR(1,K)
      IF(CRIT(K) .GT. AMIN) GO TO 40
      GO TO 39
40    AMIN=CRIT(K)
      TEST=STAR(1,K)
      NEWLNK=LLSTR(1,K)
      NEWNOD=NNSTR(1,K)
39 CONTINUE
   CHECK=LENGTH(1)+TEST
   IF(CHECK .GT. RANGE) CALL TOURNU
   IF(CHECK .GT. RANGE) GO TO 53
      IL=LH(NTOURS,NEWLNK)
      IR=RH(NTOURS,NEWLNK)
      S(NTOURS,IL)=NEWNOD
      S(NTOURS,NEWNOD)=IR
   LINKS(1)=LINKS(1)+1
53 RETURN
   END

   SUBROUTINE GRATOR

C     SUBROUTINE SELECTS NEXT CITY OR CITIES TO BE VISITED
```

```
C    AND PLACES THEM IN THE APPROPRIATE SUBTOJR

      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
     1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
      DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
      INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
      REAL LENGTH


      ISAVTR=NTOURS
      DO 71 N=1,ISAVTR
         NTOURS=N
         LIMIT=LINKS(N)
         CALL TNCRMT
         AMAX=99999.
      DO 72 J=1,KK
      DO 73 I=1,LIMIT
         IF(THETA(I,J) .LT. AMAX) GO TO 74
         GO TO 73
74       NN=NONODE(J)
         LL=I
         AMAX=THETA(I,J)
73    CONTINUE
         STAR(N,J)=AMAX
         NNSTR(N,J)=NN
         LLSTR(N,J)=LL
         AMAX=99999.
72    CONTINUE
71    CONTINUE
      NTOURS=ISAVTR
      IFLAG=0
      DO 75 I=1,NTOURS
      AMAX=99999.
      DO 76 J=1,KK
         IF(STAR(I,J) .LT. AMAX) GO TO 77
         GO TO 76
77       IN=NNSTR(I,J)
         INL=LLSTR(I,J)
         AMAX=STAR(I,J)
         ICOMPR=J
76    CONTINUE
      DO 78 K=1,NTOURS
         IF(K .EQ. I) GO TO 78
         IF(AMAX .LT. STAR(K,ICOMPR)) GO TO 78
         CHECK=STAR(K,ICOMPR)+LENGTH(K)
         IF(CHECK .LT. RANGE) GO TO 75
78    CONTINUE
         CHECK=AMAX+LENGTH(I)
         IF(CHECK .GT. RANGE) GO TO 79
         IFLAG=1
         IL=LH(I,INL)
         IR=RH(I,INL)
         S(I,IL)=IN
         S(I,IN)=IR
         LINKS(I)=LINKS(I)+1
         GO TO 75
79       STAR(I,ICOMPR)=99999.
75    CONTINUE
      IF(IFLAG .EQ. 0) CALL TOJRNU
      CALL ENDPTS
      CALL TORDIS
      CALL TOVIST
      CALL CRSEXC
      RETURN
      END

      SUBROUTINE TNCRMT
```

```
C      SUBROUTINE COMPUTES THE INCREMENTAL DISTANCE INCREASE
C      OF PLACING ANY GIVEN NODE INTO ANY GIVEN LINK OF A
C      SUBTOUR

       COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
      1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
      1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
      1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH

       IPRINT=0
       LIMIT=LINKS(NTOURS)
       DO 26 J=1,KK
       DO 27 I=1,LIMIT
          NL=LH(NTOURS,I)
          NR=RH(NTOURS,I)
          IN=NONODE(J)
          THETA(I,J)=D(NL,IN)+D(IN,NR)-D(NL,NR)
   27  CONTINUE
   26  CONTINUE

C     IF USER DESIRES TO PRINT COMPUTED VALUES OF THETA(I,J)
C     SET IPRINT=6

       IF(IPRINT .NE. 6) GO TO 50
       DO 112 I=1,LIMIT
          WRITE(6,113) (THETA(I,J),J=1,KK)
  113     FORMAT(9F10.2)
  112  CONTINUE
   50  RETURN
       END

       SUBROUTINE THREEX

C      SUBROUTINE CHECKS CITIES IN A GIVEN SUBTOUR TO
C      DETERMINE IF THE LENGTH OF THAT SUBTOUR MAY BE
C      DECREASED BY RELOCATING THE CITY IN THAT SUBTOUR

       COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
      1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
      1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
      1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH

   55  DO 60 M=1,NTOURS
          IF(LINKS(M) .LT. 4) GO TO 48
          LIMIT=LINKS(M)-1
       DO 41 I=1,LIMIT
          NL=LH(M,I)
          EXNODE=RH(M,I)
          NR=S(M,EXNODE)
          TSTVAL=D(NL,EXNODE)+D(EXNODE,NR)-D(NL,NR)
       INLIM=I+1
       NEWLIM=LINKS(M)
       DO 12 K=1,NEWLIM
          IF(K.EQ.I.OR.K.EQ.INLIM) GO TO 12
          IL=LH(M,K)
          IR=RH(M,K)
          COMVAL=D(IL,EXNODE)+D(EXNODE,IR)-D(IL,IR)
          IF(COMVAL .LT. TSTVAL) GO TO 45
   12  CONTINUE
```

```
   41 CONTINUE
   48    IF(M .EQ. NTOURS) GO TO 46
   60 CONTINUE
      GO TO 46

C    CREATE THE IMPROVED TOUR

   45 S(M,IL)=EXNODE
       S(M,EXNODE)=IR
      S(M,NL)=NR
      CALL ENDPTS
      CALL TORDIS
      GO TO 55
   46 RETURN
      END

      SUBROUTINE TOURNU

C     SUBROUTINE CREATES A NEW SUBTOUR WHEN IT IS NOT
C     FEASIBLE TO COMPLETE A VISIT TO EACH CITY WITH THE
C     PRESENT NUMBER OF SUBTOURS


      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH


      AMIN=99999.
      LIMIT=LINKS(NTOURS)
      NEWTOR=NTOURS+1
      DO 62 J=1,KK
         NOD=NONODE(J)
         IF(D(1,NOD) .LT. AMIN) GO TO 63
         GO TO 62
   63    AMIN=D(1,NOD)
         IN=NOD
   62 CONTINUE
      S(NEWTOR,1)=IN
      S(NEWTOR,IN)=1
      LINKS(NEWTOR)=2
      NTOURS=NEWTOR
      RETURN
      END

      SUBROUTINE CRSEXC

C     SUBROUTINE CHECKS CITIES IN A GIVEN SUBTOUR TO
C     DETERMINE IF TOTAL DISTANCE TRAVELED MAY BE DECREASED
C     BY PLACING THEM IN A DIFFERENT SUBTOUR


      COMMON D(25,25),S(25,25),INNODE(25,25),NONODE(25),
     1RH(25,25),LH(25,25,LINKS(25),THATA(25,25),NTOURS,
      1IEND,NC,TOTDIS,LENGTH(25),RANGE,ISTOP,PRED(25,25)
       DIMENSION X(25),Y(25),STAR(25,25),NNSTR(25,25),
     1LLSTR(25,25),SECSTR(25,25),SNNSTR(25,25),SLLSTR(25,25)
     1,CRIT(25),OLDRH(25,25),OLDLH(25,25),Z(25)
       INTEGER S,RH,OLDRH,OLDLH,EXNODE,PRED
       REAL LENGTH

   86 DO 80 I=1,NTOURS
         NOD=S(I,1)
         PRED(I,NOD)=1
   81    INT=S(I,NOD)
         PRED(I,INT)=NOD
         NOD=INT
```

```
      IF(INT .NE. 1) GO TO 81
   80 CONTINUE
      DO 82 I=1,NTOURS
      LIMIT=LINKS(I)
      DO 83 J=1,NTOURS
      IF(I .EQ. J) GO TO 83
      NUMBER=LINKS(J)-1
      IF(NUMBER .EQ. 1) GO TO 82
      DO 84 K=1,NUMBER
      DO 85 M=1,LIMIT
      NL=LH(I,M)
      NR=RH(I,M)
      IN=INNODE(J,K)
      IBFR=PRED(J,IN)
      IAFT=S(J,IN)
      THETA(M,K)=D(NL,IN)+D(IN,NR)-D(NL,NR)
      BETA=D(IBFR,IN)+D(IN,IAFT)-D(IBFR,IAFT)
      IF(THETA(M,K) .GE. BETA) GO TO 85
      CHECK=THETA(M,K)+LENGTH(I)
      IF(CHECK .GT. RANGE) GO TO 85


      S(J,IBFR)=IAFT
      S(J,IN)=0
      LINKS(J)=LINKS(J)-1

C     INSERT IN TOUR I

      S(I,NL)=IN
      S(I,IN)=NR
      LINKS(I)=LINKS(I)+1
      CALL ENDPTS
      CALL TOVIST
      CALL TORDIS
      GO TO 86
   85 CONTINUE
   84 CONTINUE
   83 CONTINUE
   82 CONTINUE
      RETURN
      END
```

LIST OF REFERENCES

1. Bellmore, M. and Hong, S., " Transformation of Multisalesmen Problem to the Standard Traveling Salesman Problem", JACM, v. 21, p. 500-504, July 1974.

2. Bellmore, M. and Nemhauser, G.L., " The Traveling Salesman Problem: A Survey", Operations Research, v. 16, p. 538-558, May-June 1968.

3. Bradley, Gordon H., " Survey of Deterministic Networks", AIIE Transactions, v. 7, p. 222-234, September 1975.

4. Case Western Reserve University Operations Research Department Technical Memorandum 182, A Model for the Development of Measures of Effectiveness in Aircraft Planning, by Alexander Brandt, June 1970.

5. Christofides, W. and Eilon, S., " An Algorithm for the Vehicle Dispatching Problem", Operations Research Quartertly, v. 20, p. 309-318, September 1969.

6. Darden, L., The Earth in the Looking Glass, p. 121-142, Doubleday, 1974.

7. Detwyler, T.R., Man's Impact on Environment, p. 684-700, McGraw Hill, 1971.

8. Drobny, N. and Dee, N., " Land: An Emerging Dimension of Planning", Research Outlook, v. 7, p. 6-10, 1975. (Research Outlook is a twice yearly publication of the Battelle Memorial Institute, 505 King Avenue, Columbus, Ohio 43201).

9. Karp, R.L., and Thompson, G.L., " AHeuristic Approach to Solving the Traveling-Salesman Problem", Management Science, v. 10, p. 225-248, January 1964.

10. Kwan, M., " Graphic Programming Using Odd or Even Points", Chinese Mathematics, v. 1, p. 273-277, 1962.

11. Little, J., Murty, K., Sweeney, D., and Karel, C., " An Algorithm for the Traveling Salesman Problem", Operations Research, v. 11, p. 972-989, November-December 1963.

12. MIT Operations Research Center Report 113, Vehicle Routin Problems: Formulations and Heuristic Solution Techniques, by Bruce L. Golden, August 1975

13. National Aeronautics and Space Administration, Earth Resources Survey Systems, v.1, 1972

14. Raymond, T.C., " Heuristic Algorithm for the Traveling Salesman Problem", IBM Journal of Research and Development, v. 13, p. 400-406, 1969.

15. Remote Sensing, National Academy of Sciences, 1970.

16. Shahrokki, F., ed., Remote Sensing of Earth Resources, p. v.1, University of Tennessee, 1972.

17. Wagner, H., Principles of Operations Research, p. 165-210, Prentice Hall, 1969.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Chief, Civil Schools Branch Department of Army US Army Military Personnel Center Alexandria, Virginia 22332	1
3.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
4.	Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
5.	Professor G.T. Howard, Code 55Hk Department of Operations Research Naval Postgraduate School Monterey, California 93940	2
6.	Professor G.H. Bradley, Code 55Bz Department of Operations Research Naval Postgraduate School Monterey, California 93940	1

7. CPT John W. Harms, USA 2
Department of Mathematics
United States Military Academy
West Point, New York 10996

8. Roger D. Arno, M.S. 240-5 2
Ames Research Center
Moffett Field, California 94035